REMARKS

This is a full and timely response to the outstanding final Office Action mailed December 6, 2006. Reconsideration and allowance of the application and pending claims are respectfully requested.

I.   Allowable Subject Matter

Applicant appreciates the Examiner's indication that claim 29 is allowable over the prior art of record. Applicant notes, however, that the Examiner also included claim 29 in the rejections under 35 U.S.C. § 102. Applicant respectfully requests that the Examiner remove claim 29 from the list of rejected claims in the next official paper.

Applicant further notes that claims 30 and 32 depend from claim 29 and therefore are likewise allowable through their dependence from claim 29. Applicant, therefore, respectfully requests that the Examiner remove claims 30 and 32 from the list of rejected claims and further expressly indicate that claims 30 and 32 are allowed.

In that it is believed that every rejection has been overcome, it is respectfully submitted that each of the claims that remains in the case is presently in condition for allowance.

II.   Claim Rejections - 35 U.S.C. § 101

Claim 35 has been rejected under 35 U.S.C. § 101 as being drawn to non-statutory subject matter. In particular, the Office Action states that claim 35 does not "require use of hardware or a tangible medium."

Applicant disagrees that claim 35 does not "require use of hardware or a tangible medium" given that the invention claimed in claim 35 is such a medium. In particular, claim 35 is directed to a "computer readable memory". Such a "memory" *is* a hardware component and clearly qualifies as a "manufacture" under 35 U.S.C. § 101. *See* 35 U.S.C. § 101. Accordingly, Applicant respectfully submits that claim 35 is drawn to statutory subject matter and respectfully requests that the rejection of claim 35 under 35 U.S.C. § 101 be withdrawn.

Regarding the Examiner's position that a "memory" cannot store a "system", Applicant notes that the term "system" does not necessarily require hardware. For instance, a suite of software could rightly be described as a "system". Regardless, Applicant submits that Applicant's use of the term "system" has no bearing on whether claim 35 is or is not directed to statutory subject matter.

III.   Claim Rejections - 35 U.S.C. § 102(e)

Claims 10, 11, 13, 35, and 37-39 have been rejected under 35 U.S.C. § 102(e) as being anticipated by *Bennett* (U.S. Pat. No. 7,114,104). Applicant respectfully traverses this rejection.

It is axiomatic that "[a]nticipation requires the disclosure in a single prior art reference of each element of the claim under consideration." *W. L. Gore & Associates, Inc. v. Garlock, Inc.*, 721 F.2d 1540, 1554, 220 U.S.P.Q. 303, 313 (Fed. Cir. 1983). Therefore, every claimed feature of the claimed invention must be represented in the applied reference to constitute a proper rejection under 35 U.S.C. § 102(e).

In the present case, not every feature of the claimed invention is represented in the Bennett reference. Applicant discusses the Bennett reference and Applicant's claims in the following.

A.   The Bennett Disclosure

Bennett discloses a process fault monitor for Unix systems. *Bennett*, column 2, lines 33-53. The fault monitor watches operating system resources associated with the processes to determine when an interruption occurs, evaluates the status of the process, and selectively signals the interruption, for example, to an event router that may in turn signal the interruption to a fault handling resource. *Id.* In some embodiments, the fault monitor uses a fault detection driver to identify the start of new processes and locate the operating system resources associated with those processes for monitoring. *Id.*

B.   Applicant's Claims

Bennett fails to teach several of Applicant's explicit claim limitations. Applicant discusses some of those claim limitations in the following.

1. Claims 10, 11, 13, and 37

Applicant's independent claim 10 provides as follows (emphasis added):

> 10. A method for controlling the execution of a child process created from a parent process, the method comprising:
> instrumenting a parent process;
> monitoring execution of the parent process with a process monitor to collect information as to the run-time behavior of the parent process;
> before a vfork system call is executed, receiving with the process monitor indicia from the parent process that a vfork system call will be executed by the parent process;
> suspending execution of the parent process;
> extracting with the process monitor a process identifier from the indicia, the process identifier identifying a child process to be generated by the parent process when the parent process executes the vfork system call;
> setting with the process monitor a process monitor thread to observe trace events generated by the child process;
> resuming execution of the parent process to enable the parent process to execute the vfork system call; and
> again suspending execution of the parent process.

As a first matter, Bennett does not teach a method for "controlling the execution of a child process". Although Bennett briefly discusses child processes that are spawned from parent processes, Bennett says nothing of "controlling the execution" of any processes, be they parent or child. Instead, Bennett is only concerned with *monitoring* processes, primarily for the purpose of identifying faults such as "abnormal terminations." Regarding column 6, lines 30-50 of the Bennett reference, which were

identified by the Examiner, those lines say nothing of controlling execution of a child process.

Second, Bennett does not teach "instrumenting" a parent process. Although Bennett generally describes the "OS fork routine 224" and the "processes 203, 240, and 250" in column 6, lines 30-50 (identified by the Examiner), nowhere is instrumenting a parent process discussed.

Third, Bennett does not teach "before a vfork system call is executed, receiving with the process monitor indicia from the parent process that a vfork system call will be executed by the parent process". Although Bennett generally discusses child processes and mentions the word "vfork( )" in column 6, lines 30-50 (again identified by the Examiner), that portion of Bennett's disclosure says nothing about a parent process indicating that a vfork system call *will be executed* by the parent process. Furthermore, Bennett does not state that such an indication is provided to any process monitor, and this includes Bennett's "fault monitor 212."

Fourth, Bennett does not teach the action of "suspending execution of the parent process". Although it is true that Bennett describes monitoring a process to identify when a fault occurs, such monitoring clearly is not equivalent to "suspending" execution. In Bennett's system, such a suspension could be monitored but not affirmatively caused.

Fifth, Bennett does not teach "extracting with the process monitor a process identifier from the indicia, the process identifier identifying a child process to be generated by the parent process when the parent process executes the vfork system call". As an initial matter, the "indicia" being described is the indicia provided from the

claimed "parent process". As noted above, Bennett fails to teach a parent process providing such an indication. Furthermore, although Bennett describes "process IDs" in column 8, lines 30-50 (identified by the Examiner), Bennett does not describe the process ID as "identifying a child process" and it certainly does not identify a child process "to be generated", i.e., that does not yet exist.

Sixth, Bennett does not teach "setting with the process monitor a process monitor thread to observe trace events generated by the child process". Although Bennett generally describes using threads to monitor processes, nowhere does Bennett state that such threads are processed "to observe trace events generated by" a "child process". Column 9, lines 40-65 of the Bennett reference, which were identified by the Examiner, do not mention observation of trace events or child events.

Seventh, Bennett does not teach "resuming execution of the parent process to enable the parent process to execute the vfork system call". Column 6, lines 30-60 of the Bennett reference, which were identified by the Examiner, generally discuss the creation of processes, not the resumption of execution of processes.

Eighth, Bennett does not teach "suspending execution of the parent process" for reasons described above.

In view of the foregoing, it is clear that Bennett fails to teach nearly each limitation of claim 10. Applicant therefore submits that claim 10 and its dependents are allowable over the prior art of record and requests that the rejections be withdrawn.

2.   Claims 35, 38, and 39

Applicant's independent claim 35 provides as follows (emphasis added):

> 35.   A computer readable memory that stores a system, the system comprising:
> a parent process configured to, before a vfork call is executed by the parent process, generate a pre-fork event that contains a process identifier of a child process that will be spawned from the parent process when the vfork system call is executed by the parent process; and
> a process monitor configured to receive the pre-fork event and process identifier before the vfork system call is executed by the parent process, suspend execution of the parent process, and generate a process monitor thread that enables observation of trace events generated by the child process.

Regarding claim 35, Bennett fails to teach a parent process configured to "generate a pre-fork event that contains a process identifier of a child process that will be spawned from the parent process" for reasons described above in relation to claim 10. Again, Bennett says nothing about a parent process providing an indication about a child process that will be created.

Furthermore, Bennett fails to teach a process monitor configured to "receive the pre-fork event and process identifier before the vfork system call is executed", "suspend execution of the parent process", or "generate a process monitor thread that enables observation of trace events generated by the child process" for reasons described above. Again, Bennett's parent processes generate no "pre-fork events" that can be received by a monitor, Bennett fails to teach the action of "suspending" execution of a

process, and Bennett speaks of no threads that enable observation of "trace events generated by" a "child process".
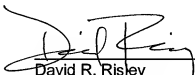
In view of the foregoing, it is clear that claim 35 and its dependents are allowable over the prior art of record. Applicant therefore requests that the rejections be withdrawn.

CONCLUSION

Applicant respectfully submits that Applicant's pending claims are in condition for allowance. Favorable reconsideration and allowance of the present application and all pending claims are hereby courteously requested. If, in the opinion of the Examiner, a telephonic conference would expedite the examination of this matter, the Examiner is invited to call the undersigned attorney at (770) 933-9500.

Respectfully submitted,

David R. Risley
Registration No. 39,345